Available online at www.sciencedirect.com
ScienceDirect
European Journal of Combinatorics 29 (2008) 72–82
European Journal of Combinatorics
www.elsevier.com/locate/ejc

Locating sensors in paths and cycles: The case of 2-identifying codes

David L. Roberts[a], Fred S. Roberts[b,1]

[a] College of Computing, Georgia Institute of Technology, Atlanta, GA 30332, USA
[b] DIMACS, Rutgers University, Piscataway, NJ 08854, USA

Received 25 January 2006; accepted 27 December 2006
Available online 23 January 2007

Abstract

For a graph G and a set $D \subseteq V(G)$, define $N_r[x] = \{x_i \in V(G) : d(x, x_i) \leq r\}$ (where $d(x, y)$ is graph theoretic distance) and $D_r(x) = N_r[x] \cap D$. D is known as an r-identifying code if for every vertex x, $D_r(x) \neq \emptyset$, and for every pair of vertices x and y, $x \neq y \Rightarrow D_r(x) \neq D_r(y)$. The various applications of these codes include attack sensor placement in networks and fault detection/localization in multiprocessor or distributed systems. Bertrand et al. [N. Bertrand, I. Charon, O. Hudry, A. Lobstein, Identifying and locating–dominating codes on chains and cycles, European Journal of Combinatorics 25 (2004) 969–987] and Gravier et al. [S. Gravier, J. Moncel, A. Semri, Identifying codes of cycles, European Journal of Combinatorics 27 (2006) 767–776] provide partial results about the minimum size of D for r-identifying codes for paths and cycles and present complete closed form solutions for the case $r = 1$, based in part on Daniel [M. Daniel, Codes identifiants, Rapport pour le DEA ROCO, Grenoble, June 2003]. We provide complete solutions for the case $r = 2$.

1. Introduction

The problem of placing sensors or detectors in a network arises in many applications including homeland security, civil engineering, manufacturing, fault detection in distributed or multiprocessor systems, etc. There are several goals in sensor placement: Rapid and accurate detection of attacks, faults, or contamination of a network, minimizing the cost of sensors used, and identification of the location of an attack or fault or contamination. If we represent

E-mail addresses: robertsd@cc.gatech.edu (D.L. Roberts), froberts@dimacs.rutgers.edu (F.S. Roberts).
[1] Tel.: +1 732 445 4303; fax: +1 732 445 5932.

0195-6698/$ - see front matter
doi:10.1016/j.ejc.2006.12.006

the network by an undirected graph, the problem of locating sensors to guarantee source identification has been formalized in several ways in the literature. This problem turns out to be NP-complete for general networks in its various formalizations and surprisingly complex for simple network topologies such as paths and cycles.

Bertrand et al. [2] study this problem for paths and cycles and provide a partial solution for detectors of varying strengths under two different formulations. The problem has been completely solved under both formulations for detectors that can detect attacks on immediate neighbors only. The main purpose of this paper is to provide the complete solution under one of the formulations when detectors can detect attacks on vertices up to two steps away in the network.

We were led to our interest in the sensor placement problems considered here by the work of Berger-Wolf et al. [1]. Their paper formalizes several sensor placement problems in networks represented by directed graphs. In particular, it formalizes the problem of attack detection and/or source identification with a fixed number of sensors and of detection and/or source identification within a given time limit. Berger-Wolf et al. argue that the complex goals of sensor placement require careful analysis in order to achieve the goals. In the closing section, we make a few remarks about our results that underscore the point that sensor placement strategies require careful analysis by methods of computer science and mathematics, since the results are sometimes unexpected.

To make our ideas precise, let $G = (V, E)$ be an undirected graph, D be a set of vertices in G at which we place detectors, and r be a positive integer. Let $N_r[x]$ be the set of all vertices in V to which there is a path of length at most r from x (so in particular $x \in N_r[x]$) and $D_r(x) = N_r[x] \cap D$. In certain literature, path length corresponds to the elapsed time before a detector is activated after an attack. In this sense, $D_r(x)$ is the set of all detectors at which an attack at x is detected in at most r time periods by some detector. We denote $D_1(x)$ by $D(x)$. We say that D is an *r-dominating set* in G if for every vertex x of G, $D_r(x) \neq \emptyset$, i.e., there is path of length $\leq r$ from x to some $y \in D$. (A 1-dominating set is of course a dominating set.) We say that D is an *r-identifying code* in G (*r-IC*) if it is an r-dominating set and if whenever $x \neq y$ are vertices, $D_r(x) \neq D_r(y)$. In an r-IC, the set of detectors activated by an attack provides a unique signature that allows us to determine where the attack took place. We shall seek the smallest d so that there is an r-IC of d vertices in G, if there is one. If so, we denote d by $M_r^I(G)$. If $r = 1$, we drop r in our terminology and speak of an identifying code or IC, and use $M^I(G)$ to mean $M_1^I(G)$. Identifying codes have been studied by many authors, starting with Karpovsky et al. [19] and motivated by fault detection in multiprocessor networks. An r-identifying code enables a central controller to identify inoperable processors in a multiprocessor network. Some of the processors have "monitors" that report to the central controller if some processor within distance r is inoperable. If the monitors have been placed so that they define an r-identifying code, then the central controller can determine which processor is inoperable based on the reports by the monitors.

Note that not every graph has an r-identifying code. For instance, in the complete graph, every $D(x)$ is the set of all vertices. More generally, in any graph, if there are two vertices with the same closed neighborhood, there can be no 1-identifying code.

A closely related concept is defined as follows. We say that a set D of vertices in graph G is an *r-locating–dominating set* or *r-LD set* for short if for all $x \notin D$, $D_r(x) \neq \emptyset$ and for all $x, y \notin D, x \neq y$, we have $D_r(x) \neq D_r(y)$. The smallest d such that there is an r-LD set of size d is denoted by $M_r^{LD}(G)$. If $r = 1$, we speak of locating–dominating sets, LD sets and $M^{LD}(G)$. Locating–dominating sets were introduced (for $r = 1$) by Slater [23], motivated by

nuclear power plant safety. *r-locating–dominating* sets can also be used for fault detection in distributed systems. Note that in contrast to r-identifying codes, r-LD sets always exist, since the entire vertex set of a graph is an r-LD set.[2]

The literature about r-identifying codes and r-locating–dominating sets has become quite extensive. See [20] for a recent bibliography with over 100 entries. In this paper, we will limit ourselves to r-identifying codes.

One reason for our interest in paths and cycles is that paths are appropriate in applications like subway tunnels and cycles in applications like airport tram loops, to give two examples. (Note that while the trains or trams may only go one way, contaminants or pathogens can go either way through tunnels.) Many other interesting topologies have been investigated in the literature. Karpovsky et al. [19] study r-identifying codes in specific topologies of interest in distributed computing, in particular binary cubes, nonbinary cubes, various meshes, and trees, and various others study r-identifying codes and r-LD sets for binary hypercubes, e.g., Blass et al. [3]. Cohen et al. [10,12] and Honkala and Lobstein [18] study square lattices while Cohen et al. [8,11] and Karpovsky et al. [19] study hexagonal and triangular grids. Carson [4], Rall and Slater [21] and Slater [23] study complete multipartite graphs and planar and outerplanar graphs.

As noted above, problems of finding optimal r-identifying sets or r-LD sets are difficult. Berger-Wolf et al. [1] show that for directed graphs the problem of minimizing the size of an r-identifying code is NP-complete. NP-completeness results for directed graphs for both r-LD sets and r-identifying codes were also obtained by Charon et al. [6,7] for both digraphs and undirected graphs, and for identifying codes by Cohen et al. [9] and for LD sets by Colbourn et al. [13], both for undirected graphs.[3] By way of contrast, Slater [22] gives a linear time algorithm for finding optimal LD sets in acyclic graphs, in particular trees, and Colbourn et al. [13] give a linear time dynamic programming algorithm for finding optimal LD sets in series–parallel graphs.

In Section 2 we summarize the values of $M^I(G)$ for paths and cycles, that is in the case where detectors can only detect attacks one step away in a network. Section 3 finds the values $M_2^I(G)$ for paths and cycles, i.e., it analyzes the case where we have stronger detectors, ones that can detect attacks up to two steps away. Finally, Section 4 includes closing remarks.

2. Identifying codes for paths and cycles

In this section, we summarize the known results for $M^I(P_n)$ and $M^I(C_n)$ where P_n is the path of n vertices and C_n is the cycle of n vertices.

The following theorem was proven for the case of even cycles by Bertrand et al. [2] and for odd cycles by Daniel [14]. It is also proven for odd cycles by Gravier et al. [16]. Its analogue for LD sets was proven by Slater [23].

Theorem 2.1. *For the cycle C_n:*

(1) $M^I(C_4) = 3$, $M^I(C_{2k}) = k$, $k \geq 3$;
(2) $M^I(C_5) = 3$, $M^I(C_{2k+1}) = k + 2$, $k \geq 3$.

[2] A variant of an r-locating–dominating set is defined by Carson and Oellermann [5]. Let D be a set of vertices $\{v_1, v_2, \ldots, v_k\}$ in G and for each $x \notin S$, let $\vec{x}$ be the vector whose ith entry is $\min\{r + 1, d(x, v_i)\}$ where $d(u, v)$ is the distance from u to v in the graph. Then we say that D is an *r-reference–dominating set* if for all $x \notin D$, $D_r(x) \neq \emptyset$ and for all $x, y \notin D$, $\vec{x} = \vec{y}$ iff $x = y$. If $r = 1$, a 1-reference–dominating set is the same as a 1-locating–dominating set.

[3] Carson and Oellermann [5] give similar NP-completeness results for reference–dominating sets.

The following theorem was first proven by Bertrand et al. [2]. Its analogue for LD sets is due to Slater [23].

Theorem 2.2. *For the path P_n:*

(1) *$M^I(P_2)$ is undefined, $M^I(P_{2k}) = k+1, k \geq 2$;*
(2) *$M^I(P_{2k+1}) = k+1, k \geq 0$.*

3. 2-Identifying codes for paths and cycles

We observed in Section 2 that if two vertices in G have the same closed neighborhood, then there can be no IC. Recall that $N_2[x] = \{y : d(x, y) \leq 2\}$, where $d(x, y)$ is graph theoretic distance. Then if there are two vertices with the same N_2, there can be no 2-IC. This shows that P_3, P_4, C_4, and C_5 have no 2-IC.

In the following, we assume that the vertices of P_n or C_n have been labeled consecutively as $x_1, x_2, \ldots, x_n$. When we are dealing with a cycle, we also use addition and subtraction modulo n, so that, for example, x_{5n+4} means x_4.

3.1. 2-ICs for cycles

Lemma 3.1. *Suppose graph G has maximum degree 2, $y_1, y_2, y_3, y_4, y_5, y_6$ is a path in G, and D is a 2-IC for G. Then it is not possible to have $y_1 \notin D$ and $y_6 \notin D$.*

Proof. If $y_1 \notin D$ and $y_6 \notin D$, then $D_2(y_3) = D_2(y_4)$. □

Lemma 3.2. *If $n > 6$, D is a 2-IC for a cycle C_n iff*

(1) *there are no six consecutive vertices with the first and last not in D,*
(2) *there are no five consecutive vertices none of which is in D.*

Proof. Condition (2) is necessary and sufficient for the condition that $D_2(x) \neq \emptyset$ for all x. Necessity of condition (1) follows from Lemma 3.1. We shall now observe sufficiency of conditions (1) and (2) for the condition $D_2(x) \neq D_2(y), x \neq y$. Consider x_i and $x_j, i < j$, and assume (A) that the distance from x_i to x_j is no larger in a clockwise direction around the cycle than in a counterclockwise direction. To show that $D_2(x_i) \neq D_2(x_j)$ if $i+1 \leq j \leq i+5$, apply condition (1) and (A) to x_{i-2} and x_{i+3}. If $j > i+5$, apply condition (2) and (A) to $x_{i-2}, x_{i-1}, x_i, x_{i+1}, x_{i+2}$. □

Parts of the next theorem were known previously. In particular, (1) was proven by Bertrand et al. [2], who show that the same result holds for arbitrary r. (1) is also proven by Gravier et al. [16]. Result (2)(c) follows from a more general result of Gravier et al. [16], namely that if $2r+1$ divides n odd, then $M_r^I(C_n) = \frac{(n+1)}{2} + r$. For arbitrary r, Bertrand et al. and Gravier et al. also provide lower bounds and Gravier et al. give exact values for some special cases. For $r = 2$, Bertrand et al. give exact values of $M_2^I(C_n)$ when $n = 10k+1$. Bertrand et al. also give results about optimal r-LD sets for cycles for arbitrary r.

Theorem 3.3. *For the cycle C_n:*

(1) *$M_2^I(C_4)$ is undefined, $M_2^I(C_6) = 5$, $M_2^I(C_{2k}) = k$ for $k \geq 4$.*
(2) *$M_2^I(C_5)$ is undefined and if $k = 5p+q, q \in \{0, 1, 2, 3, 4\}$, then*

(a) $M_2^I(C_{2k+1}) = k+2$ *if* $q = 0$ *and* $p > 0$;
(b) $M_2^I(C_{2k+1}) = k+1$ *if* $q = 1$ *and* $p > 0$;
(c) $M_2^I(C_{2k+1}) = k+3$ *if* $q = 2$ *and* $p > 0$;
(d) $M_2^I(C_{2k+1}) = k+1$ *if* $q = 3$ *and* $p \geq 0$;
(e) $M_2^I(C_{2k+1}) = k+2$ *if* $q = 4$ *and* $p > 0$, *with* $M_2^I(C_9) = 5$.

Proof. We have already observed that C_4 has no 2-IC. For C_6, we note that including all but x_6 gives a 2-IC, so $M_2^I(C_6) \leq 5$. Suppose D is a 2-IC with at most four vertices. Assume without loss of generality that $x_1 \notin D$. Then using Lemma 3.1 in both the clockwise and counterclockwise direction implies that $x_2 \in D, x_6 \in D$. If $x_4 \notin D$, then $D_2(x_1) = D_2(x_4)$. Thus, either $x_3 \notin D$ or $x_5 \notin D$, without loss of generality the former. But then $D_2(x_4) = D_2(x_6)$. We conclude that $M_2^I(C_6) \geq 5$, so $M_2^I(C_6) = 5$.

Lemma 3.2 implies the constraint $x_i \in D \vee x_{i+5} \in D$ for $i = 1, 2, \ldots, n$.[4] There are n such constraints and each x_i is a term in exactly two of them. Thus, if D has d vertices, at most $2d$ such disjunctions are satisfied. It follows that D must have at least $\lceil \frac{n}{2} \rceil$ vertices. If $n = 2k$, then D must have at least k vertices, so $M_2^I(C_{2k}) \geq k$. If $n = 2k+1$, then similarly $M_2^I(C_{2k+1}) \geq k+1$. If $k \geq 4$, the set of x_i for i odd is a 2-IC for C_{2k}, so $M_2^I(C_{2k}) \leq k$, which completes the proof for C_{2k}.

We now turn to C_{2k+1}. We return to the constraints above and, for notational convenience, abbreviate x_j by j and abbreviate $x_i \in D$ or $x_j \in D$ by $i \vee j$. Choose $i \in \{1, 2, 3, 4, 5\}$. Consider the following stream of constraints, which we call *constraint stream* i:

$$i \vee i+1 \times 5, \quad i+1 \times 5 \vee i+2 \times 5, \quad \ldots, \quad i+(g_i-1) \times 5 \vee i+g_i \times 5, \\ i+g_i \times 5 \vee h_i,$$

where

$$i + g_i \times 5 \leq 2k+1 < i+(g_i+1) \times 5 \equiv h_i (\mathrm{mod}[2k+1]), \quad h_i \in \{1, 2, 3, 4, 5\}.$$

Suppose $k = 5p+q$, $q \in \{0, 1, 2, 3, 4\}$. If $k \neq 5p+2$, then $h_1 \neq 1$. Then constraint stream 1 leads into constraint stream h_1, which leads into constraint stream h_{h_1}, and so on until we hit every stream and end with the last $h_i = 1$. Putting the streams together in this order gives us the *full constraint stream*. For example, if $k = 13$, constraint stream 3 is given by

$$3 \vee 8, \qquad 8 \vee 13, \qquad 13 \vee 18, \qquad 18 \vee 23, \qquad 23 \vee 1.$$

This can be abbreviated as $3\|8\|13\|18\|23\|1$, where the notation means that of every two adjacent vertices, at least one is in D. We still refer to the condition on two adjacent vertices as a constraint. Using the same notation, the full constraint stream is

$$1\|6\|11\|16\|21\|26\|4\|9\|14\|19\|24\|2\|7\|12\|17\|22\|27\|5\|10\|15\|20\|25\|3\|8\|13\|18\|23\|1$$

Generalizing from $k = 13$, we note that if $k = 5p+3$, we have $2k+1 = 10p+7$, so $g_3 = 2p$ since $3 + 2p \times 5 \leq 10p+7 < 3+(2p+1) \times 5$. If $k = 5p+2$, then $h_i = i$ and there is no full constraint stream. By way of contrast, if $k = 5p+4$, then $g_3 = 2p+1$.

We first prove (2)(b). We already know that $M_2^I(C_{2k+1}) \geq k+1$. To show that $M_2^I(C_{2k+1}) \leq k+1$, construct D as follows. Among the vertices $1, 2, \ldots, 10$, choose 3, 4, 6, 7, 10 to be in D.

[4] A similar condition for arbitrary r corresponds to the idea of a transversal in an auxiliary graph $C'_{(n,r)}$ [16].

Call this a "pattern" and repeat this pattern by choosing 13, 14, 16, 17, 20 from the vertices 11, 12, ..., 20, and so on through the first $10p$ vertices. Finally, take $2k+1 = 10p+3$ and 1 in D. Due to the "cyclic" structure of D, one immediately checks that (1) and (2) of Lemma 3.2 hold and that D has $5p+2 = k+1$ vertices.

This construction was derived by observing that since the full stream has $2k+1$ constraints and each vertex is in exactly two constraints, then if we can satisfy all the constraints with $k+1$ vertices of D, there must be exactly one constraint where both vertices are in D and all other constraints have exactly one of their vertices in D. Which vertex in a constraint is in D is forced upon us by traversing the full stream starting from the constraint with both vertices in D. Without loss of generality this constraint is $1 \vee 1+1\times 5$. All other constructions of D in this paper were obtained using similar reasoning.

The proof of (2)(d) is analogous. For D of $5p+4 = k+1$ vertices and satisfying Lemma 3.2, use the pattern 2, 5, 6, 8, 9 through the first $10p$ vertices and add $10p+2$, $10p+5$, $10p+6$ and 1.

We turn next to the proof of (2)(e). We assume that we can find a 2-IC D of $k+1$ vertices and shall reach a contradiction. As in the explanation of the construction in (2)(b), once we take 1 and $1+1\times 5$ in D (without loss of generality), the rest of the membership of D is forced upon us:

- from stream 1: use vertices 1 and $1+z\times 5$, z odd [a total of $p+2$ vertices];
- then from stream 2: vertices $2+z\times 5$, z odd [$p+1$ additional vertices];
- then from stream 3: vertices $3+z\times 5$, z odd [$p+1$ additional vertices];
- then from stream 4: vertices $4+z\times 5$; z odd [$p+1$ additional vertices];
- finally from stream 5: vertices $5+z\times 5$, z odd [p additional vertices].

This satisfies condition (1) of Lemma 3.2 and uses $5p+5 = k+1$ vertices. However, condition (2) of Lemma 3.2 is violated if $p>0$, since then 11, 12, 13, 14, 15 are not in D. We conclude that $M_2^I(C_{2k+1}) \geq k+2$ if $k = 5p+4$, $p>0$. If $p=0$, the construction gives us $D = \{1,6,7,8,9\}$ and it is easy to show that this is a 2-IC of size $k+1$, so it is optimal. This shows that $M_2^I(C_9) = 5$.

To complete the proof for $k = 5p+4$, we construct a 2-IC with $k+2$ vertices. We do this by taking the following $5p+6 = k+2$ vertices for D: Use the pattern 3, 6, 7, 9, 10 through the first $10p$ vertices and add vertices $10p+3$, $10p+6$, $10p+7$, $10p+8$, $10p+9$, 1. It is easy to check that Lemma 3.2 holds.

The proof of (2)(a) is similar. Suppose $k = 5p$, $p>0$, and there is a 2-IC D of $k+1$ vertices. Without loss of generality, we take 1 and $1+1\times 5$ in D and the rest of the membership of D is forced upon us. Since both 1 and 6 are in D, we find that $7, 8, 9, 10, 11 \notin D$. Thus, condition (2) of Lemma 3.2 is violated. We construct D with $5p+2 = k+2$ vertices and satisfying Lemma 3.2 by using the pattern 2, 3, 4, 6, 10 on the first $10p$ vertices and adding vertices 1 and $10p+1$.

We now turn to Condition 2(c) and assume $k = 5p+2$. In this case, each constraint stream begins and ends in i and the vertices in the different constraint streams are disjoint. Thus, to satisfy Condition (2) of Lemma 3.2, we need to satisfy all of the constraints in each stream i separately. Since each stream has $2p+1$ constraints, we need at least $p+1$ vertices from it to be put into D in order to satisfy all constraints. Thus, we need at least $5(p+1) = k+3$ vertices in D, which shows that $M_2^I(C_{2k+1}) \geq k+3$ in this case. It remains to prove that we can find a 2-IC D with $5p+5 = k+3$ vertices. We do this as follows: Use the pattern 2, 4, 6, 8, 10 on the first $10p$ vertices, and add the vertices 1, 3, 5, $10p+2$, and $10p+4$. This completes the proof of Theorem 3.3. □

3.2. 2-ICs of paths

We turn now to paths. The next lemma will help us to calculate $M_2^I(P_n)$.

Lemma 3.4. *Consider a path P_n with vertices $x_1, x_2, \ldots, x_n$ in order and suppose D is a 2-IC for P_n. Then:*

(1) $x_4 \in D$ *and* $x_{n-3} \in D$,
(2) $x_5 \in D$ *and* $x_{n-4} \in D$.

Proof. If $x_4 \notin D$, then $D_2(x_1) = D_2(x_2)$, and similarly for x_{n-3}. If $x_5 \notin D$, then $D_2(x_2) = D_2(x_3)$, and similarly for x_{n-4}. □

We now have:

Lemma 3.5. *If $n > 6$, D is a 2-IC for a path P_n iff the following conditions hold:*

(1) *there are no six consecutive vertices with the first and last not in D,*
(2) *there are no five consecutive vertices none of which is in D,*
(3) $x_4, x_5, x_{n-3}, x_{n-4} \in D$,
(4) x_1, x_2, *or* $x_3 \in D$ *and* x_n, x_{n-1}, *or* $x_{n-2} \in D$.

Proof. Necessity of condition (1) follows by Lemma 3.1, of condition (2) since otherwise $D_2(x) = \emptyset$ for the middle vertex x, of condition (3) by Lemma 3.4, and of condition (4) because $D_2(x_1) \neq \emptyset$, $D_2(x_n) \neq \emptyset$. Sufficiency follows by a proof analogous to that of Lemma 3.2. □

Lemma 3.5 allows us to proceed for a path P_n much as we did with cycles. Constraint streams are again the focus of our argument but since we have paths and not cycles, we modify the definition of constraint stream i to omit the last disjunction $i + g_i \times 5 \vee h_i$. We will consider the cases $n = 5p + q$, $q \in \{0, 1, 2, 3, 4\}$. For example, if $n = 5p + 1$, the constraint stream 3 is given by

$$3, 3 + 1 \times 5, 3 + 2 \times 5, \ldots, 3 + (p - 2) \times 5, 3 + (p - 1) \times 5.$$

Since the constraint streams have disjoint sets of vertices, we are in a situation analogous to that of cycles in the case $k = 5p + 2$ where we considered satisfying the constraints in each stream separately.

We summarize the results in the following theorem. The result in case (2) for p even is proven by Bertrand et al. [2], where lower bounds are also given for M_r^I for arbitrary r that in fact match the exact results given in the theorem in cases (1), (2), and (3) for p even and cases (4) and (5) for p odd. Bertrand et al. also give results for r-LD sets for paths and credit I. Honkala with obtaining the exact values for $M_r^{LD}(P_n)$ when $r = 2$.[5]

Theorem 3.6. *Let $n = 5p + q$, $q \in \{0, 1, 2, 3, 4\}$.*

(1) *If $q = 0$, $p \geq 1$, then $M_2^I(P_n) = \frac{5p}{2} + 1$ if p is even, $M_2^I(P_n) = \frac{5(p-1)}{2} + 4$ if p is odd.*
(2) *If $q = 1$, $p \geq 1$ then $M_2^I(P_n) = \frac{5p}{2} + 1$ if p is even, $M_2^I(P_n) = \frac{5(p-1)}{2} + 5$ if p is odd.*

[5] Let $M_r^{RD}(G)$ be the size of the smallest r-reference–dominating set in G (as defined in an earlier footnote). The exact values of $M_r^{RD}(P_n)$ are derived by Carson and Oellermann [5].

(3) *If* $q = 2$, $p \geq 1$, *then* $M_2^I(P_n) = \frac{5p}{2} + 2$ *if* p *is even,* $M_2^I(P_n) = \frac{5(p-1)}{2} + 5$ *if* p *is odd. Also,* $M_2^I(P_2)$ *is undefined.*

(4) *If* $q = 3$, $p \geq 1$, *then* $M_2^I(P_n) = \frac{5p}{2} + 3$ *if* p *is even,* $M_2^I(P_n) = \frac{5(p-1)}{2} + 5$ *if* p *is odd. Also,* $M_2^I(P_3)$ *is undefined.*

(5) *If* $q = 4$, $p \geq 1$, *then* $M_2^I(P_n) = \frac{5p}{2} + 3$ *if* p *is even,* $M_2^I(P_n) = \frac{5(p-1)}{2} + 5$ *if* p *is odd. Also,* $M_2^I(P_4)$ *is undefined.*

Proof. As in the proof of Theorem 3.3, we use i as an abbreviation for vertex x_i.

(1) If $i \in \{1, 2, 3, 4, 5\}$, the constraint stream i is given as follows:

$$i, \quad i + 1 \times 5, \quad i + 2 \times 5, \quad \ldots, \quad i + (p-2) \times 5, \quad i + (p-1) \times 5.$$

By Lemma 3.5, condition (3), we know that $4, 5, 5p - 3, 5p - 4 \in D$. This tells us that the first constraint in streams 4 and 5 is satisfied and the last constraint in streams 1 and 2 by choosing detectors at these locations. To satisfy Condition (4) of Lemma 3.5, there are three possible cases: (1A) choose 3 and $5p - 2$ from stream 3; (1B) choose 1 from stream 1 and $5p - 2$ from stream 3, 2 from stream 2 and $5p - 2$ from stream 3, 3 from stream 3 and $5p - 1$ from stream 4, 3 from stream 3 and $5p$ from stream 5; (1C) choose 1 from stream 1 and $5p - 1$ from stream 4, 1 from stream 1 and $5p$ from stream 5, 2 from stream 2 and $5p - 1$ from stream 4, 2 from stream 2 and $5p$ from stream 5. (We lump these cases because counting number of detectors needed is the same in each of the situations in each case.)

Consider first case (1A). We need to satisfy all of the constraints in stream 1 and we have already placed $5p - 4$ in the detector set, satisfying the last constraint. There are $p - 2$ remaining constraints, none containing $5p - 4$, so we need at least $\left\lceil \frac{p-2}{2} \right\rceil$ detectors to satisfy the remaining constraints. Turning to stream 2, since we have already placed $5p - 3$ in the detector set, satisfying the last constraint, and since $5p - 3$ does not appear in other constraints in this stream, the number of remaining constraints once again require at least $\left\lceil \frac{p-2}{2} \right\rceil$ detectors. The same number at least is required for streams 4 and 5. In stream 3, we need to use detectors 3 and $5p - 2$, thus satisfying the first and last constraints, and at least $\left\lceil \frac{p-3}{2} \right\rceil$ detectors are needed to satisfy the remaining $p-3$ constraints, none of which contain either 3 or $5p - 2$. It follows that we need at least

$$4 + 2 + \left\lceil \frac{p-3}{2} \right\rceil + 4 \times \left\lceil \frac{p-2}{2} \right\rceil$$

detectors in all. If p is even, this number is $\frac{5p}{2} + 1$, whereas if p is odd, the number is $\frac{5(p-1)}{2} + 5$.

We need to do a similar analysis in Case (1B). We consider the situation where we choose 1 from stream 1 and $5p - 2$ from stream 3. After choosing $1, 4, 5, 5p - 2, 5p - 3, 5p - 4$ we need at least $\left\lceil \frac{p-3}{2} \right\rceil$ detectors to satisfy the remaining $p - 3$ constraints in stream 1, and, in each of the other streams, at least $\left\lceil \frac{p-2}{2} \right\rceil$ detectors to satisfy the remaining $p-2$ constraints. Thus, we need at least $\frac{5p}{2} + 1$ detectors if p is even and at least $\frac{5(p-1)}{2} + 5$ if p is odd.

Last, we consider Case (1C) and suppose we choose 1 from stream 1 and $5p-1$ from stream 4. After choosing $1, 4, 5, 5p - 1, 5p - 3, 5p - 4$, to satisfy the remaining constraints, we need at least $\left\lceil \frac{p-3}{2} \right\rceil$ for stream 1, at least $\left\lceil \frac{p-2}{2} \right\rceil$ for stream 2, at least $\left\lceil \frac{p-1}{2} \right\rceil$ for stream 3,

at least $\left\lceil \frac{p-3}{2} \right\rceil$ for stream 4, and at least $\left\lceil \frac{p-2}{2} \right\rceil$ for stream 5. Thus, the number of detectors needed in all is $\frac{5p}{2} + 2$ if p is even and $\frac{5(p-1)}{2} + 4$ if p is odd.

Finally, comparing the required minimum number of detectors in all three cases, we see that when p is even, the minimum is $\frac{5p}{2} + 1$, which is achieved in both cases (1A) and (1B), and when p is odd, the minimum is $\frac{5(p-1)}{2} + 4$, which is achieved in Case (1C) only. This establishes these values as lower bounds on $M_2^I(P_n)$ in the case $n = 5p$.

We next establish these values as upper bounds. Consider first the case where p is even. As in the proof of Theorem 3.3, if $p = 2s$, we build a set D of $\frac{5p}{2} + 1 = 5s + 1$ elements and satisfying the conditions of Lemma 3.5, as follows: Start with the pattern 1, 4, 5, 7, 8 on the first $5p$ vertices and add the number $5p - 4$.

Next, suppose p is odd. If $p = 2s + 1$, build D of $\frac{5(p-1)}{2} + 4 = 5s + 4$ elements and satisfying the conditions of Lemma 3.5 as follows: Use the pattern 1, 4, 5, 7, 8 on the first $5(p - 1)$ vertices and add the vertices $5p - 4, 5p - 3, 5p - 1, 5p$.

The proofs of parts (2) through (5) are analogous and we leave the details to the reader. We simply include below the instructions for how to achieve an optimal detector set D in each case.

(2) p even: use pattern 1, 4, 5, 7, 8 on first $5p$ and add $5p + 1$.
p odd: use pattern 2, 4, 5, 6, 8 on first $5(p - 1)$ and add $5p - 3, 5p - 2, 5p - 1, 5p, 5p + 1$.

(3) p even: use pattern 1, 4, 5, 7, 8 on first $5p$ and add $5p - 1, 5p + 1$.
p odd: use pattern 2, 4, 5, 6, 8 on first $5(p - 1)$ and add $5p - 2, 5p - 1, 5p, 5p + 1, 5p + 2$.

(4) p even: use pattern 1, 4, 5, 7, 8 on first $5p$ and add $5p - 1, 5p, 5p + 1$.
p odd: use pattern 1, 4, 5, 7, 8 on first $5(p - 1)$ and add $5p - 4, 5p - 1, 5p, 5p + 2, 5p + 3$.

(5) p even: use pattern 1, 4, 5, 7, 8 on first $5p$ and add $5p, 5p + 1, 5p + 4$.
p odd: use pattern 2, 4, 5, 6, 8 on first $5(p - 1)$ and add $5p - 3, 5p - 1, 5p, 5p + 1, 5p + 3$. □

4. Closing remarks

The emphasis in this paper has been on determining exact values for $M^I(G)$ and $M_2^I(G)$ for given graphs, in particular paths and cycles. It should be noted that our results for paths and cycles are obtained by explicit constructions that yield simple algorithms that are linear in time in terms of the number of vertices. We have given results for 1-ICs and 2-ICs. It would be of interest to extend them to r-ICs for $r > 2$. It would also be of interest to completely describe $M_2^{LD}(G)$ for cycles. Bertrand et al. [2] give the exact value of $M_2^{LD}(G)$ for paths (and credit the result to I. Honkala) and for cycles if $n = 6k, k \geq 1$. Other cases remain open.

Our results give a few conclusions that, at least at first, seem somewhat surprising. Carson [4] pointed out the surprising result that increasing the range of detectors may lead us to require more detectors. He found a tree T so that $M_2^{LD}(T) = 6$ while $M_6^{LD}(T) > 6$. Our results show that using detectors with greater range can also sometimes require more detectors for perfect discrimination. In particular,

$$M_2^I(C_{10p+5}) > M^I(C_{10p+5}), \quad p \geq 1.$$

A similar result holds for paths:

$$M_2^I(P_n) > M^I(P_n), \quad n = 5, 6, 7, 10p + 3, 10p + 5, 10p + 6, 10p + 7, p \geq 1.$$

It is also worth noting that

$$M^I(C_{2k+1}) > M^I(C_{2k+2}), \quad k > 2.$$

Thus, if we take the same topology and make it longer, we may require fewer detectors. A similar result holds for M_2^I:

$$M_2^I(C_n) > M_2^I(C_{n+1}), \quad n = 6, 11, 10p + 5, 10p + 9, 10p + 11, p \geq 1.$$

These observations underscore the point made by Berger-Wolf et al. [1] that methods of computer science and mathematics are needed to analyze sensor placement strategies since the results can be unexpected.

In our formulation of the problem, we do not explicitly consider the time allocated for movement over an edge although it is implicitly considered to be equal for every edge. Considering the same problem for edges with weights representing possibly different times of movement over edges is also of interest. This problem is studied from an algorithmic point of view by Berger-Wolf et al. [1].

Our formulation of the problem is only concerned with detecting attacks at single vertices. It would be useful to formulate and solve similar perfect detection discrimination problems if we allow attacks at multiple locations. This problem is studied by Gravier and Moncel [15] and Karpovsky et al. [19] and elsewhere for identifying codes, but not for paths or cycles.

The problem with sensor failures allowed would also be of interest. Some preliminary results with one sensor failure are presented by Slater [24] while some for multiple sensor malfunctions by Honkala et al. [17].

Finally, we have limited the discussion to detection problems on networks where both detectors and attacks can only take place at vertices. The problem is also of interest and can benefit from precise analysis of the type in this paper if we weaken these restrictions and allow attacks and/or detectors anywhere along an edge.

Acknowledgements

The first author's research was performed while on appointment as a US Department of Homeland Security (DHS) Fellow under the DHS Scholarship and Fellowship Program, a program administered by the Oak Ridge Institute for Science and Education (ORISE) for DHS through an interagency agreement with the US Department of Energy (DOE). ORISE is managed by Oak Ridge Associated Universities under DOE contract number DE-AC05-00OR22750. All opinions expressed in this paper are the author's and do not necessarily reflect the policies and views of DHS, DOE, or ORISE. The second author thanks the National Science Foundation for its support under grant EIA-0205116 to Rutgers University. Both authors thank Tanya Berger-Wolf, Steve Hedetniemi, Olivier Hudry, Ortrud Oellermann, and Peter Slater for pointing them to relevant literature. Additionally, they wish to thank an anonymous reviewer for helpful suggestions and simplifications in the presentation.

References

[1] T. Berger-Wolf, W.E. Hart, J. Saia, Discrete sensor placement problems in distribution networks, Mathematical and Computer Modelling 42 (2005) 1385–1396.

[2] N. Bertrand, I. Charon, O. Hudry, A. Lobstein, Identifying and locating–dominating codes on chains and cycles, European Journal of Combinatorics 25 (2004) 969–987.

[3] U. Blass, I. Honkala, S. Litsyn, Bounds on identifying codes, Discrete Mathematics 241 (2001) 119–128.

[4] D.I. Carson, On generalized location-domination, in: Y. Alavi, A. Schwenk (Eds.), Graph Theory, Combinatorics and Applications, Wiley, New York, 1995, pp. 161–179.
[5] D.I. Carson, O.R. Oellermann, A generalized reference–domination parameter. Unpublished manuscript, Department of Computer Science, University of Natal, 1995.
[6] I. Charon, O. Hudry, A. Lobstein, Identifying and locating–dominating codes: NP-completeness results for directed graphs, IEEE Transactions on Information Theory IT-48 (2002) 2192–2200.
[7] I. Charon, O. Hudry, A. Lobstein, Minimizing the size of an identifying or locating–dominating code in a graph is NP-hard, Theoretical Computer Science 290 (2003) 2109–2120.
[8] G. Cohen, S. Gravier, I. Honkola, A. Lobstein, M. Mollard, C. Payan, G. Zemor, Improved identifying codes for the grid, Electronic Journal of Combinatorics 6 (1999) R19.
[9] G. Cohen, I. Honkala, A. Lobstein, G. Zemor, On identifying codes, in: A. Barg, S. Litsyn (Eds.), Codes and Association Schemes, in: DIMACS Series, vol. 56, American Mathematical Society, Providence, RI, 2001, pp. 97–109.
[10] G. Cohen, I. Honkala, A. Lobstein, G. Zemor, New bounds for codes identifying vertices in graphs, The Electronic Journal of Combinatorics 6 (1999) R14.
[11] G. Cohen, I. Honkala, A. Lobstein, G. Zemor, Bounds for codes identifying vertices in the hexagonal grid, SIAM Journal on Discrete Mathematics 13 (2000) 492–504.
[12] G. Cohen, I. Honkala, A. Lobstein, G. Zemor, On codes identifying vertices in the two-dimensional square lattice with diagonals, IEEE Transactions on Computers 50 (2001) 174–176.
[13] C.J. Colbourn, P.J. Slater, L.K. Stewart, Locating–dominating sets in series–parallel networks, in: Proceedings of the 16th Annual Conference on Numerical Mathematics and Computing, Manitoba, Winnipeg, 1986, pp. 135–162.
[14] M. Daniel, Codes identifiants, Rapport pour le DEA ROCO, Grenoble, June 2003.
[15] S. Gravier, J. Moncel, Construction of codes identifying sets of vertices, Electronic Journal of Combinatorics 12 (2005) R13.
[16] S. Gravier, J. Moncel, A. Semri, Identifying codes of cycles, European Journal of Combinatorics 27 (2006) 767–776.
[17] I. Honkala, T. Laihonen, S. Ranto, On locating–dominating codes in binary hamming spaces, Discrete Mathematics and Theoretical Computer Science 6 (2004) 265–282.
[18] I. Honkala, A. Lobstein, On the density of identifying codes in the square lattice, Journal of Combinatorial Theory Series B 85 (2002) 297–306.
[19] M.G. Karpovsky, K. Chakrabarty, L.B. Levitin, On a new class of codes for identifying vertices in graphs, IEEE Transactions on Information Theory 44 (1998) 599–611.
[20] A. Lobstein, Codes identifiants et localisateurs-dominateurs dans les graphes: Une bibliographie http://www.infres.enst.fr/lobstein/bibLOCDOMetID.html, November 2005.
[21] D.F. Rall, P.J. Slater, On location-domination numbers for certain classes of graphs, Congressus Numerantium. 45 (1984) 97–106.
[22] P.J. Slater, Domination and location in acyclic graphs, Networks 17 (1987) 55–64.
[23] P.J. Slater, Dominating and reference sets in a graph, Journal of Mathematical and Physical Sciences 22 (1988) 445–455.
[24] P.J. Slater, Fault-tolerant locating–dominating sets, Discrete Mathematics 249 (2002) 179–189.